All of the pending claims (i.e., 1-20, 34-50 and 62-77) are rejected under 35 U.S.C. §103 as being obvious over Ma in view of Ha. These rejections are respectfully traversed.

I. Overview of the Present Invention and Cited References

Many computer systems include one or more host computers and one or more storage systems that provide logical volumes (which the host computer perceives to correspond to an actual physical storage device) to the host computer. (see Applicants' specification at page 1, lines 6-7 and page 2, lines 10-14). Most host computers also include an application layer that refers to data objects used thereby with an identifier (such as a file name) that provides no information about where the file is physically stored among the logical volumes presented by the storage system. (page 2, lines 8-10). Typically, the host computer also includes a file system and/or logical volume manager layer (LVM) that maps each data object specified by the application layer to particular logical volume locations. (page 2, lines 10-13).

In a conventional computer system, the label or identifier used by the file system/LVM layer to refer to a logical volume is dependent upon the physical configuration of the computer system, and may include information that describes a particular physical configuration path through which the logical volume is accessible. (page 2, lines 14-23). These logical volume identifiers are assigned when the computer system is brought on line, and the host computer is booted or initialized to go through a device discovery process wherein a unique identifier is assigned to each logical volume. (page 3, lines 21-25).

Events can occur that may change a physical configuration of a computer system. For example, a port on the storage system or host computer may fail, such that the system may need to be physically reconfigured so that one or more logical volumes are made available to the host over a different path. (page 4, lines 7-11). After the physical reconfiguration, the host computer would typically need to be rebooted to recognize the reconfigured nature of the computer system (page 4, lines 11-13). It should be appreciated that in doing so, the device identifier that is assigned to one or more logical volumes and is dependent upon the physical configuration of the computer system may change, such that a logical volume may have a new identifier after reconfiguration. (page 4, lines 13-19). As mentioned above, the file system/LVM layer

executing on the host computer accesses logical volumes using the device identifiers that, in a conventional system, are dependent upon the physical configuration of the system. Thus, when a physical reconfiguration takes place and those identifiers are altered, it could cause significant problems for applications executing on the host computer. (page 4, lines 20-27). Specifically, there is a risk that after reconfiguration, the file system/LVM layer will attempt to access logical volumes using the old (i.e., prior to reconfiguration) identifiers, which can result in the wrong data being accessed for a read or write. (page 5, lines 4-11).

Conventional computer systems do not provide any technique for automatically detecting changes in system configuration which can lead to the above-described problems. (page 5, lines 12-14). Thus, to avoid such problems, a system administrator must manually intervene to appreciate that a reconfiguration has occurred, and must take appropriate action to avoid the above-described problems. In accordance with one embodiment of the present invention, a technique is provided for automatically detecting changes in the system configuration, and for determining the manner in which the resources are to be accessed by the host computer in the new configuration.

Another disadvantage with conventional computer systems is that even if a system administrator accurately detects that a physical reconfiguration has taken place, conventional systems do not provide any technique for dynamically addressing such system configuration changes, and require that the system be rebooted or reinitialized to recognize the changes that have taken place. In accordance with another embodiment of the present invention, a technique is provided for dynamically reconfiguring a computer system, without reinitializing a host computer or application program thereon to alter a manner in which the application program accesses a computer system resource.

The above-described aspects of the present invention can be implemented in any of numerous ways, and are not limited to any particular implementation technique. In accordance with one embodiment of the present invention, the host computer is provided with information that identifies resources available to it (e.g., logical volumes) in a manner that is independent of the physical configuration of the computer system. This information can then be correlated or mapped to information that identifies the resources (e.g., logical volumes) in a manner that is

dependent upon the physical configuration of the computer system. This mapped information can be used to detect changes in the configuration of the computer system, and to determine the manner in which resources are accessed according to a newly detected configuration. Furthermore, the mapping information can be employed to achieve a dynamic reconfiguration of the computer system when a configuration change is detected.

The foregoing summary of the invention is provided merely to assist the Examiner in appreciating various aspects of the present invention. However, this summary does not apply to each of the independent claims, and the language of the independent claims may differ in material respects from the summary provided above. Thus, Applicants respectfully request that careful consideration be given to the language of each of the independent claims and that each be addressed on its own merits, without relying on the summary provided above. In this respect, Applicants do not rely on the summary provided above to distinguish any of the claims of the present invention over the prior art. Rather, Applicants rely only upon the arguments provided below.

Ma

Ma is directed to a technique for updating a distributed object database application without shutting down the application (col. 4, lines 12-14). In this respect, Ma teaches that certain objects can be modified and then automatically recompiled and reloaded without completely shutting down the application. For example, if an additional field is to be added to a database, Ma teaches that the other parts of the database application can continue to operate while the new field is added. (col. 4, lines 17-20).

The sections of Ma highlighted in the Office Action describe a process for changing a distributed application, as shown in Fig. 4 of Ma. Initially, changes to the application are defined by a user using a user-interface (col. 7, lines 4-5). Examples of the types of changes described include adding, deleting or modifying fields in a database. (col. 7, lines 6-7). Once the changes are defined, a server re-generates source code for the changed objects, which are then re-complied and re-linked. (col. 7, lines 15-18). The newly modified classes are then loaded into the server, and any existing objects that belong to the changed class are marked as invalid. (col.

7, lines 22-24). In this manner, the distributed application (e.g., a database) is updated without taking it offline.

Ha

Ha teaches a method for remotely updating programmed instructions. Specifically, Ha teaches remotely upgrading the basic input-output system (BIOS), which is a set of instructions typically stored on a personal computer's ROM chip that handles its input-output functions. The BIOS facilitates communication between the operating system and the computer's hardware to control such basic operations as input via a keyboard or mouse, or access to memory.

Ha teaches upgrading the BIOS of a personal computer from a host computer which transfers a modified BIOS image to the personal computer using serial communication. (col. 4, lines 7-10 and 25-27). The conventional BIOS stored in the BIOS ROM is removed, and the received BIOS image is written into the BIOS ROM, which is a writable flash PROM (col. 3, lines 60-61 and col. 4, lines 29-34). The upgrade is completed by rebooting the computer. (column 5, lines 23-24).

II. The Combination of Ma and Ha Under §103 Is Improper

The Office Action asserts that one of ordinary skill in the art would have been motivated to incorporate the BIOS accessing method as taught by Ha into the upgradeable system of Ma "because Ma operates with methods to access mobile objects in a distributed system and Ha suggested that optimization can be obtained with access methods." Applicants respectfully disagree.

Ma is related to a system for updating a distributed application (such as a database) in a manner that does not require re-initialization, whereas Ha is directed to a technique for upgrading a BIOS and specifically teaches that following the upgrade the computer must be rebooted. (col. 5, lines 23-24). The techniques for updating a distributed application such as a database are different from those employed in updating a BIOS, such that it is respectfully asserted that one of ordinary skill in the art would not have been motivated to make any modification whatsoever to Ma based upon the teachings of Ha. Furthermore, since the Ha

updating technique requires re-initialization, one of ordinary skill in the art would not have been motivated to employ the Ha technique in the system of Ma, as it would frustrate the express teachings of Ma that the updating of the application be performed without re-initialization.

In view of the foregoing, it is respectfully asserted that the combination of Ma and Ha under 35 U.S.C. §103 is improper, such that the rejection of all of Applicants' claims under §103 as being obvious over these references should be withdrawn.

III. The Claims Patentably Distinguish Over the Combination

A. Claims Direct to Aspect of the Present Invention Relating to Determining A Different Configuration For A Computer System and The Manner in Which A Computer System Resource is Accessed in The Different Configuration

Claims 1-13

Claim 1 is directed to a method of responding to changes in a computer system configuration impacting a manner in which at least one computer system resource is accessed by a host computer, comprising (A) storing information relating to a first configuration of the computer system at a first point in time, the first configuration relating to a first manner of accessing at least one computer system resource by the host computer; (B) determining a second configuration of the computer system at a second point in time, the second configuration relating to a second manner of accessing the at least one computer system resource by the host computer; (C) comparing the second configuration with the first configuration to determine whether the second configuration differs from the first configuration; and (D) when it is determined that the second configuration differs from the first, determining the second manner of accessing the at least one computer system resource by the host computer.

Neither Ma nor Ha teaches comparing a second computer system configuration with a first configuration to determine whether the second configuration differs from the first configuration as recited in claim 1. The Office Action contends that Ma teaches such a comparison at col. 7, lines 19-21. However, the cited passage merely demonstrates that Ma implements code changes defined by a user, and does not teach that a comparison occurs between new and old versions of the code. At col. 7, lines 13-22, Ma states:

> Once a user has finished defining the changes, these changes are sent to the server for updating the application. From the user's definitions, the server re-generates the source code for changed class definitions of objects, step 48. The object class definitions are re-compiled and re-linked, step 50. Some of the existing object instances now no longer match their modified object class definitions. Notification is made from the server that some object classes have changed, step 52. The newly modified classes are loaded to the server, using a cache of classes.

As should be appreciated from the foregoing, in the system of Ma, changes to the database are defined by a user, and then the techniques disclosed therein are employed to implement those changes. There is simply no teaching in Ma of comparing the versions of code at two different points in time to determine whether they are different.

Ha also fails to teach comparison of first and second versions of BIOS to determine whether they are different. Rather, the user determines when to implement a new BIOS, and Ha merely teaches a technique for modifying the BIOS by removing the old one. (see e.g., col. 5, lines 17-24). Ha does not teach comparing two versions of BIOS to determine whether they differ.

As should be appreciated from the foregoing, neither Ma nor Ha teaches the aspect of the present invention recited in claim 1 wherein first and second configurations of a computer system are compared to determine whether they differ. Therefore, it is respectfully asserted that the Office Action fails to set forth a prima facie case of obviousness with respect to claim 1, as it does not demonstrate how the combination of Ma and Ha teaches all of the limitations recited in claim 1. It is respectfully asserted that claim 1 patentably distinguishes over the prior art of record, such that the rejection of claim 1 under §103 should be withdrawn.

Claims 2-13 depend from claim 1 and are patentable for at least the same reasons.

Claims 34-44

Independent claim 34 is directed to a computer-readable medium that, when executed on a host computer, performs a method substantially similar to the method recited in claim 1. Therefore, for the reasons set forth above with respect to claim 1, claim 34 patentably

distinguishes over the prior art of record, such that the rejection of claim 34 under 35 U.S.C. §103 should be withdrawn.

Claims 35-44 depend from claim 34 and are patentable for at least the same reasons.

Claims 62-71

Independent claim 62 is directed to a host computer comprising, inter alia, first determining means for determining a second configuration of a computer system at a second point in time and comparing means for comparing the second configuration with a first configuration to determine whether they differ. As should be appreciated from the foregoing, neither Ma nor Ha teaches means for determining a configuration of a computer system, nor means for comparing two configurations to determine whether they differ. Thus, it is respectfully asserted that the Office Action fails to set forth a prima facie case of obviousness with respect to claim 62, as it fails to demonstrate how all of the limitations of claim 62 are met by the prior art. Therefore, it is respectfully requested that the rejection of claim 62 under 35 U.S.C. §103 be withdrawn.

Claims 63-71 depend from claim 62 and are patentable for at least the same reasons.

B. Claims Directed to The Embodiment of the Present Invention Relating to Dynamic Reconfiguration

Claims 14-20

Claim 14 is directed to a method of reconfiguring a computer system which includes a host computer and at least one computer system resource accessible to at least one application program executing on the host computer. The method comprises a step of dynamically reconfiguring the computer system, without re-initializing the host computer or the application program, to alter a manner in which the at least one application program accesses the at least one computer system resource.

In ¶10, the Office Action asserts that Ma teaches updating without re-initialization, and that "the above claim limitations are obvious in view of the combination" for that reason. Applicants respectfully disagree, and note that the Office Action fails to explain how the

combination is believed to meet each of the limitations of claim 14, such that no prima facie case of obviousness has been set forth.

Ma does not teach a method of reconfiguring a computer system as alleged in the Office Action. As discussed above, Ma merely teaches the updating of an application, such as a database. Updating an application does not constitute reconfiguration of a computer system.

Furthermore, claim 14 further recites dynamically reconfiguring the computer system to alter a manner in which an application program accesses at least one computer system resource. Ma teaches an example of updating a database application to include an additional field. Ma does not teach dynamically reconfiguring a computer system to alter the manner in which an application program accesses at least one computer system resource as recited in claim 14.

In view of the foregoing, it is respectfully asserted that the rejection of claim 14 under §103 is improper, and should be withdrawn. Claims 15-20 depend from claim 14 and are patentable for at least the same reasons.

Claims 45-50

Independent claim 45 is directed to a computer-readable medium that, when executed on a host computer, performs a method substantially similar to that recited in claim 14. Therefore, for the reasons set forth above, claim 45 patentably distinguishes over the prior art of record, such that the rejection of claim 45 under 35 U.S.C. §103 should be withdrawn.

Claims 46-50 depend from claim 45 and are patentable for at least the same reasons.

Claims 72-76

Claim 72 is directed to a host computer comprising at least one controller to dynamically reconfigure a computer system, without re-initializing the host computer or an application program thereon, to alter a manner in which the at least one application program accesses at least one computer system resource.

As discussed above in connection with claim 14, the prior art of record does not teach or suggest a host computer including at least one controller to dynamically reconfigure a computer system to alter a manner in which an application program accesses at least one computer system

resource. Therefore, it is respectfully asserted that the Office Action fails to set forth a prima facie case obviousness with respect to claim 72, such that the rejection of claim 72 under 35 U.S.C. §103 is improper and should be withdrawn.

Claims 73-76 depend from claim 72 and are patentable for at least the same reasons.

CONCLUSION

In view of the foregoing remarks, it is believed that this application is now in condition for allowance. A notice to this effect is respectfully requested. If it is believed that this application is not in condition for allowance for any reason, the Examiner is requested to contact the undersigned at the telephone number listed below to discuss any outstanding issues relating to the allowability of the application.

Respectfully submitted,
Steven M. Blumenau et al., Applicants

By: Richard Giunta

Richard F. Giunta, Reg. No. 36,149
Wolf, Greenfield & Sacks, P.C.
600 Atlantic Avenue
Boston, Massachusetts 02210-2211
Telephone: (617) 720-3500

Docket No. E00295.70087.US
Date: November 6, 2002
x12/12/02x